SIAM J. SCI. COMPUT.
Vol. 32, No. 4, pp. 1929–1949

ITERATIVE NEAR-FIELD PRECONDITIONER FOR THE MULTILEVEL FAST MULTIPOLE ALGORITHM*

LEVENT GÜREL† AND TAHİR MALAS‡

Abstract. For iterative solutions of large and difficult integral-equation problems in computational electromagnetics using the multilevel fast multipole algorithm (MLFMA), preconditioners are usually built from the available sparse near-field matrix. The exact solution of the near-field system for the preconditioning operation is infeasible because the LU factors lose their sparsity during the factorization. To prevent this, incomplete factors or approximate inverses can be generated so that the sparsity is preserved, but at the expense of losing some information stored in the near-field matrix. As an alternative strategy, the entire near-field matrix can be used in an iterative solver for preconditioning purposes. This can be accomplished with low cost and complexity since Krylov subspace solvers merely require matrix-vector multiplications and the near-field matrix is sparse. Therefore, the preconditioning solution can be obtained by another iterative process, nested in the outer solver, provided that the outer Krylov subspace solver is flexible. With this strategy, we propose using the iterative solution of the near-field system as a preconditioner for the original system, which is also solved iteratively. Furthermore, we use a fixed preconditioner obtained from the near-field matrix as a preconditioner to the inner iterative solver. MLFMA solutions of several model problems establish the effectiveness of the proposed nested iterative near-field preconditioner, allowing us to report the efficient solution of electric-field and combined-field integral-equation problems involving difficult geometries and millions of unknowns.

Key words. preconditioning, fast multipole method (FMM), multilevel fast multipole algorithm (MLFMA), sparse approximate inverse preconditioners, flexible solvers, variable preconditioning, integral equations, computational electromagnetics

AMS subject classifications. 31A10, 65F10, 78A45, 78M05

DOI. 10.1137/09076101X

1. Introduction.

Real-life problems in computational electromagnetics (CEM) are large, requiring not only substantial computing resources but also fast solvers. The multilevel fast multipole algorithm (MLFMA) is one such popular method widely used for the solution of Helmholtz-type scattering and radiation problems. MLFMA is used in combination with a Krylov subspace solver to reduce the computational and memory requirements of a dense matrix-vector product to $\mathcal{O}(n \log n)$ [9]. Hence, large problems can be solved with modest computing resources [18]. However, the success of the employed iterative method is mainly determined by the preconditioner, especially when the linear system to be solved is ill-conditioned.

Preconditioners can be broadly classified as one of two types: forward (or implicit) and inverse (or explicit). Forward preconditioning (implicit) refers to finding an easily invertible operator $\overline{\boldsymbol{M}}$ for the system $\overline{\boldsymbol{A}} \cdot \boldsymbol{x} = \boldsymbol{b}$, while $\overline{\boldsymbol{M}}$ approximates $\overline{\boldsymbol{A}}$ in some

*Received by the editors June 3, 2009; accepted for publication (in revised form) March 15, 2010; published electronically July 6, 2010. This work was supported by the Scientific and Technical Research Council of Turkey (TUBITAK) under research grants 105E172 and 107E136, the Turkish Academy of Sciences in the framework of the Young Scientist Award Program (LG/TUBA-GEBIP/2002-1-12), and by contracts from ASELSAN and SSM.

http://www.siam.org/journals/sisc/32-4/76101.html

†Department of Electrical and Electronics Engineering, Bilkent University, TR-06800, Bilkent, Ankara, Turkey (lgurel@bilkent.edu.tr).

‡Department of Electrical and Electronics Engineering and Computational Electromagnetics Research Center (BiLCEM), Bilkent University, TR-06800, Bilkent, Ankara, Turkey (tmalas@ee.bilkent.edu.tr).

sense. Then, instead of the original system, the preconditioned system

$$\overline{\boldsymbol{M}}^{-1} \cdot \overline{\boldsymbol{A}} \cdot \boldsymbol{x} = \overline{\boldsymbol{M}}^{-1} \cdot \boldsymbol{b} \tag{1.1}$$

is solved. For inverse preconditioning (explicit), $\overline{\boldsymbol{M}}$ directly approximates the inverse of the system matrix, and the preconditioned system becomes

$$\overline{\boldsymbol{M}} \cdot \overline{\boldsymbol{A}} \cdot \boldsymbol{x} = \overline{\boldsymbol{M}} \cdot \boldsymbol{b}. \tag{1.2}$$

The idea is based on the observation that as $\overline{\boldsymbol{M}}$ approximates $\overline{\boldsymbol{A}}$, the product $\overline{\boldsymbol{M}}^{-1} \cdot \overline{\boldsymbol{A}}$ approximates the identity matrix (for forward preconditioners), and convergence can be attained in fewer iterations.

In (1.1) and (1.2), we apply *left preconditioning*. We can also apply *right preconditioning*, in which we should solve the systems

$$\overline{\boldsymbol{A}} \cdot \overline{\boldsymbol{M}}^{-1} \cdot \boldsymbol{y} = \boldsymbol{b}, \qquad \boldsymbol{x} = \overline{\boldsymbol{M}}^{-1} \cdot \boldsymbol{y} \tag{1.3}$$

and

$$\overline{\boldsymbol{A}} \cdot \overline{\boldsymbol{M}} \cdot \boldsymbol{y} = \boldsymbol{b}, \qquad \boldsymbol{x} = \overline{\boldsymbol{M}} \cdot \boldsymbol{y} \tag{1.4}$$

for forward and inverse preconditioning, respectively.

In this paper, we concentrate on effective preconditioning of large linear systems of equations generated with the electric-field integral equation (EFIE) and the combined-field integral equation (CFIE). CFIE is a linear combination of EFIE and the magnetic-field integral equation (MFIE). The application domain of MFIE is restricted to geometries with closed surfaces. Since CFIE includes MFIE, the use of CFIE is also limited to closed-surface problems. Hence, for geometries with open surfaces, EFIE is the only choice from these three integral equations. However, EFIE produces highly ill-conditioned systems.

When such systems are solved with MLFMA, only the near-field interactions are kept in memory, which constitutes a sparse portion of the dense coefficient matrix. To achieve a strong preconditioner, the information provided by the near-field matrix should be effectively used. Using the exact factorization of the near-field matrix for preconditioning is too expensive because of fill-ins. One common way to overcome the problem is to use an incomplete factorization of the sparse near-field matrix, which is the procedure used in the most-established preconditioning methods, namely, incomplete LU (ILU) methods [3]. In our previous work [31], we showed that, among various ILU preconditioners, incomplete LU preconditioner with threshold (ILUT) [36] and its pivoted version (ILUTP) [37] are successful in CEM problems and produce iteration counts that are very close to those obtained by the exact factorization of the near-field matrix. However, because ILU methods are inherently sequential, in parallel MLFMA implementations, sparse-approximate-inverse (SAI) preconditioners must be used instead [7, 30].

On the other hand, it is widely observed that SAI is not as successful as ILU in reducing the number of iterations and the solution times [4]. Therefore, in order to make better use of the near-field matrix as a preconditioner, we propose solving the near-field system with a nested iterative solver. We also use SAI as the preconditioner of this inner iterative solver. Then we show that only a few inner iterations suffice to obtain strong preconditioners. In this way, both the iteration counts and the solution times are greatly reduced, compared to the case where SAI is used alone. This nested

preconditioning scheme can of course also be used with other fixed preconditioners to accelerate the inner iterative solver. We note that one should use flexible solvers for the solution of the original (outer) system if the preconditioning operator is not fixed, as it is in this scheme.

In the next section, we summarize the integral equations employed in this study in order to describe the origins of the matrix equations to be solved. In sections 3 and 4, we outline discretization of the integral equations and MLFMA, respectively. Then, in section 5, we comment on the preconditioning of systems generated by integral equations, and we detail the proposed preconditioning method. Section 6 presents the numerical results and comparisons of the iterative near-field preconditioner with SAI. We summarize our conclusions in section 7.

2. Surface integral equations. Surface integral equations are extensively used in CEM for solving scattering and radiation problems [34, 33, 38]. Integral-equation formulations can be obtained by defining equivalent currents on the surface of an arbitrary three-dimensional (3-D) geometry and applying boundary conditions. Various integral-equation formulations can be derived by employing different sets of boundary conditions and the testing procedure [42]. In this work, we consider the most commonly used EFIE and CFIE formulations.

2.1. The electic-field integral equation. EFIE is based on a physical boundary condition which states that the total tangential electric field vanishes on a conducting surface. Mathematically, EFIE can be expressed as

$$\hat{\boldsymbol{t}} \cdot \int_{S'} d\boldsymbol{r'}\, \overline{\boldsymbol{G}}(\boldsymbol{r}, \boldsymbol{r'}) \cdot \boldsymbol{J}(\boldsymbol{r'}) = \frac{i}{k\eta} \hat{\boldsymbol{t}} \cdot \boldsymbol{E}^{inc}(\boldsymbol{r}), \tag{2.1}$$

where $\boldsymbol{E}^{inc}(\boldsymbol{r})$ represents the incident electric field, S' is the surface of the object, $\hat{\boldsymbol{t}}$ is any tangential unit vector on S', $\boldsymbol{J}(\boldsymbol{r'})$ is the unknown induced current residing on the surface, and $\eta = \sqrt{\mu/\epsilon}$ is the intrinsic impedance of the medium. In (2.1), $\overline{\boldsymbol{G}}(\boldsymbol{r}, \boldsymbol{r'})$ is the dyadic Green's function defined as

$$\overline{\boldsymbol{G}}(\boldsymbol{r}, \boldsymbol{r'}) = \left[\overline{\boldsymbol{I}} + \frac{\nabla\nabla}{k^2}\right] g(\boldsymbol{r}, \boldsymbol{r'}), \tag{2.2}$$

where

$$g(\boldsymbol{r}, \boldsymbol{r'}) = \frac{e^{ik|\boldsymbol{r}-\boldsymbol{r'}|}}{4\pi|\boldsymbol{r}-\boldsymbol{r'}|} \tag{2.3}$$

is the scalar Green's function for the 3-D scalar Helmholtz equation. The scalar Green's function represents the response at the observation point $\boldsymbol{r}$ due to a point source located at $\boldsymbol{r'}$. In (2.1), (2.2), and (2.3), k denotes the wavenumber ($k = 2\pi\lambda$, where λ is the wavelength).

EFIE belongs to the class of first-kind integral equations, which have a weakly-singular kernel. Due to the weak singularity of the kernel, the integral equation acts as a smoothing operator and provides high accuracy with low-order basis functions, such as the commonly used Rao–Wilton–Glisson (RWG) basis functions [34]. On the other hand, because of the weak singularity of the kernel, matrices obtained with the discretization of EFIE tend to be ill-conditioned [8, 43].

To gain more information about the spectral properties of EFIE, Figure 2.1 shows the eigenvalues and the pseudospectra for a sphere problem with 930 unknowns. The

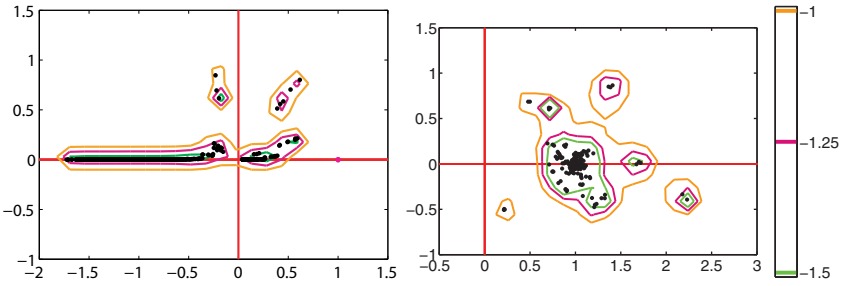

FIG. 2.1. *Pseudospectra of an* (a) *EFIE matrix and* (b) *its preconditioned version at the perturbation levels of* $10^{-1.5}$, $10^{-1.25}$, *and* 10^{-1}. *The eigenvalues are shown by black dots.*

pseudospectrum represents the topology of the eigenvalues of the perturbed matrices associated with the exact matrix [39]. From the unpreconditioned version depicted in Figure 2.1(a), we recognize two important facts. First, even with a small perturbation, the EFIE matrix becomes singular, verifying that it is ill-conditioned. Second, some eigenvalues are scattered in the left half-plane, and this is an unfavorable situation for iterative solvers. Hence, even the most robust solvers, such as the full generalized minimal residual (GMRES) method, do not converge without preconditioning, particularly for large problems. However, the preconditioned spectrum shown in Figure 2.1(b) reveals that convergence can be attained in a few iterations by employing the preconditioner that will be described in section 5.3.

2.2. The combined-field integral equation. Using the boundary condition for the tangential magnetic field on a conducting surface, MFIE can be expressed as

$$-\boldsymbol{J}(\boldsymbol{r}) + \hat{\boldsymbol{n}} \times \int_{S'} d\boldsymbol{r'} \boldsymbol{J}(\boldsymbol{r'}) \times \nabla' g(\boldsymbol{r}, \boldsymbol{r'}) = -\hat{\boldsymbol{n}} \times \boldsymbol{H}^{inc}(\boldsymbol{r}), \tag{2.4}$$

where $\hat{\boldsymbol{n}}$ is any unit normal vector on S' and $\boldsymbol{H}^{inc}(\boldsymbol{r})$ is the incident magnetic field. In (2.4), note that the boundary condition for the magnetic field is tested via the unit normal vector $\hat{\boldsymbol{n}}$. This is necessary to obtain stable solutions using a Galerkin scheme [42].

Unlike EFIE, MFIE is a second-kind integral equation that leads to diagonally dominant and well-conditioned matrices [24]. However, due to the singularity of its kernel, the accuracy of MFIE is significantly lower than that of EFIE [25, 43, 17, 13]. The identity term that results from the $\boldsymbol{J}(\boldsymbol{r})$ term in (2.4) is also another source for error [19].

CFIE is a more accurate second-kind integral equation than MFIE. It is obtained by linearly combining EFIE and MFIE, i.e.,

$$\text{CFIE} = \alpha \text{EFIE} + (1 - \alpha)\text{MFIE}, \tag{2.5}$$

where α is a parameter between 0 and 1. It is shown that $\alpha = 0.2$ or $\alpha = 0.3$ yields minimum iteration counts [15]. Among the three integral equations considered in this study, CFIE is the only formulation that is free from internal-resonance problems [24]. Furthermore, CFIE leads to well-conditioned systems, particularly for simple objects [40]. Currently, the solution of a sphere problem involving more than 200 million unknowns has been reported, where the solution is obtained in only 25 iterations with

a simple block-diagonal preconditioner (BDP) [20]. On the other hand, CFIE is not applicable to open geometries since it contains MFIE. Therefore, CFIE is preferred to MFIE for closed geometries, but EFIE, which produces ill-conditioned linear systems, particularly for large problems [19], is the mandatory choice for geometries with open surfaces.

3. Discretization of the integral equations. Following a simultaneous discretization of the integral-equation formulations and geometry surfaces, electromagnetics problems involving complicated targets can be discretized and solved numerically. In this section, we present the details of the discretization procedures.

3.1. Method of moments. We can convert the surface integral equations described in section 2 to dense linear systems using the method of moments (MOM). Using a linear operator $\mathcal{L}$, these integral equations can be denoted as

$$\mathcal{L}\{\boldsymbol{J}\} = \boldsymbol{G}, \tag{3.1}$$

where $\boldsymbol{G}$ is one of the known right-hand-side (RHS) vectors in (2.1) or (2.5). Projecting (3.1) onto the N-dimensional space span$\{\boldsymbol{j}_1, \boldsymbol{j}_2, \ldots, \boldsymbol{j}_N\}$ formed by the divergence-conforming RWG basis functions, we obtain

$$\langle \boldsymbol{j}_m, \mathcal{L}\{\boldsymbol{J}\}\rangle = \langle \boldsymbol{j}_m, \boldsymbol{G}\rangle, \qquad m = 1, 2, \ldots N, \tag{3.2}$$

where

$$\langle \boldsymbol{f}, \boldsymbol{g}\rangle = \int d\boldsymbol{r}\, \boldsymbol{f}(\boldsymbol{r}) \cdot \boldsymbol{g}(\boldsymbol{r}) \tag{3.3}$$

denotes the inner product of two vector functions $\boldsymbol{f}$ and $\boldsymbol{g}$. Then, adopting Galerkin's approach, we expand the unknown current using the same set of basis functions, i.e.,

$$\boldsymbol{J} \approx \sum_{n=1}^{N} x_n \boldsymbol{j}_n. \tag{3.4}$$

Hence, the coefficient vector $\boldsymbol{x}$ becomes the solution of the $N \times N$ linear system

$$\overline{\boldsymbol{A}} \cdot \boldsymbol{x} = \boldsymbol{b}, \tag{3.5}$$

where

$$\left(\overline{\boldsymbol{A}}\right)_{mn} = \langle \boldsymbol{j}_m, \mathcal{L}\{\boldsymbol{j}_n\}\rangle, \quad (\boldsymbol{b})_m = \langle \boldsymbol{j}_m, \boldsymbol{G}\rangle, \quad m, n = 1, 2, \ldots N. \tag{3.6}$$

A matrix entry $(\overline{\boldsymbol{A}})_{mn}$ defined in (3.6) can be interpreted as an electromagnetic interaction between the mth testing function and the nth basis function.

The RWG basis functions are defined on planar triangles. Therefore, surfaces of CEM problems are meshed accordingly using planar triangles. Each RWG basis function is associated with an edge; hence the number of unknowns for a problem becomes equal to the total number of edges in the mesh, except for the boundary edges of an open surface.

3.2. Discretization of EFIE. After the discretization of EFIE defined in (2.1) with MOM, the matrix entries can be derived as

$$\begin{aligned}\left(\overline{\boldsymbol{A}}^{EFIE}\right)_{mn} = &\int_{S_m} d\boldsymbol{r}\ \boldsymbol{t}_m(\boldsymbol{r}) \cdot \int_{S_n} d\boldsymbol{r'}\ \boldsymbol{b}_n(\boldsymbol{r'}) g(\boldsymbol{r}, \boldsymbol{r'}) \\ &-\frac{i}{k^2} \int_{S_m} d\boldsymbol{r}\ \boldsymbol{t}_m(\boldsymbol{r}) \cdot \int_{S_n} d\boldsymbol{r'}\ \boldsymbol{b}_n(\boldsymbol{r'}) \cdot [\nabla\nabla' g(\boldsymbol{r}, \boldsymbol{r'})],\end{aligned} \tag{3.7}$$

where $\boldsymbol{t}_m$ denotes a testing function and $\boldsymbol{b}_n$ denotes a basis function. Due to the double differentiation of the scalar Green's function, EFIE is highly singular in this form. However, using the divergence-conforming feature of RWG basis functions, it is possible to distribute the two differential operators onto the basis and testing functions and obtain [34]

$$\begin{aligned}\left(\overline{\boldsymbol{A}}^{EFIE}\right)_{mn} &= ik\int_{S_m} d\boldsymbol{r}\ \boldsymbol{t}_m(\boldsymbol{r})\cdot\int_{S_n} d\boldsymbol{r'}\ \boldsymbol{b}_n(\boldsymbol{r'})g(\boldsymbol{r},\boldsymbol{r'})\\ &\quad -\frac{i}{k^2}\int_{S_m} d\boldsymbol{r}\ \nabla\cdot\boldsymbol{t}_m(\boldsymbol{r})\int_{S_n} d\boldsymbol{r'}\ \nabla'\cdot\boldsymbol{b}_n(\boldsymbol{r'})g(\boldsymbol{r},\boldsymbol{r'}).\end{aligned} \tag{3.8}$$

The outer integrals in (3.8) can be evaluated numerically by employing Gaussian quadrature rules [11]. The inner integrals can be evaluated as

$$\int_{S_n} d\boldsymbol{r'}\left\{\begin{array}{c}1\\ x'\\ y'\end{array}\right\} g(\boldsymbol{r},\boldsymbol{r'}) = I_1 + I_2, \tag{3.9}$$

where

$$I_1 = \frac{1}{4\pi}\int_{S_n} d\boldsymbol{r'}\left\{\begin{array}{c}1\\ x'\\ y'\end{array}\right\}\frac{\exp(ikR)-1}{R} \tag{3.10}$$

and

$$I_2 = \frac{1}{4\pi}\int_{S_n} d\boldsymbol{r'}\left\{\begin{array}{c}1\\ x'\\ y'\end{array}\right\}\frac{1}{R}. \tag{3.11}$$

For I_1, an adaptive integration method or a Gaussian quadrature rule can be used [12]. Furthermore, for accurate computations, singularity extraction techniques are employed by sufficiently subtracting the singular parts of the integrands. The integral I_2 can be evaluated analytically [22, 28].

3.3. Discretization of CFIE. Since CFIE is a linear combination of EFIE and MFIE, both formulations should be discretized to form CFIE. The discretization of MFIE in (2.4) with RWG basis functions and a Galerkin scheme leads to

$$\begin{aligned}\left(\overline{\boldsymbol{A}}^{MFIE}\right)_{mn} &= -\int_{S_m} d\boldsymbol{r}\ \boldsymbol{t}_m(\boldsymbol{r})\cdot\boldsymbol{b}_n(\boldsymbol{r'})\\ &\quad +\int_{S_m} d\boldsymbol{r}\ \boldsymbol{t}_m(\boldsymbol{r})\cdot\hat{\boldsymbol{n}}\times\int_{S_n} d\boldsymbol{r'}\ \boldsymbol{b}_n(\boldsymbol{r'})\times\nabla' g(\boldsymbol{r},\boldsymbol{r'}).\end{aligned} \tag{3.12}$$

Since the second term in the RHS of (3.12) contains a singularity, we perform an efficient singularity extraction technique for the outer integral [25]. After the singularity extraction, (3.12) becomes

$$\begin{aligned}\left(\overline{\boldsymbol{A}}^{MFIE}\right)_{mn} &= -\frac{1}{2}\int_{S_m} d\boldsymbol{r}\ \boldsymbol{t}_m(\boldsymbol{r})\cdot\boldsymbol{b}_n(\boldsymbol{r'})\\ &\quad +\int_{S_m,PV} d\boldsymbol{r}\ \boldsymbol{t}_m(\boldsymbol{r})\cdot\hat{\boldsymbol{n}}\times\int_{S_n} d\boldsymbol{r'}\ \boldsymbol{b}_n(\boldsymbol{r'})\times\nabla' g(\boldsymbol{r},\boldsymbol{r'}),\end{aligned} \tag{3.13}$$

where PV indicates the principal value of the integral. The double integral in the second RHS term of (3.13) can be modified as [28]

$$\int_{S_m} d\boldsymbol{r}\Big(\boldsymbol{t}_m(\boldsymbol{r}) \times \hat{\boldsymbol{n}}\Big) \cdot \boldsymbol{b}_n(\boldsymbol{r}) \times \int_{PV,S_n} d\boldsymbol{r'}\ \nabla' g(\boldsymbol{r}, \boldsymbol{r'}). \tag{3.14}$$

Note that only the principal values are required for (3.14) since the the limit part is extracted. Nonetheless, the singularity extraction is applied again to smooth the integrand before an adaptive integration. The inner integral in (3.14) can be calculated as

$$\int_{PV,S_n} d\boldsymbol{r'}\ \nabla' g(\boldsymbol{r}, \boldsymbol{r'}) = I_1 + I_2 + I_3, \tag{3.15}$$

where

$$I_1 = \frac{1}{4\pi} \int_{PV,S_n} d\boldsymbol{r'}\ \nabla' \left(\frac{\exp(ikR) - 1 + 0.5k^2R^2}{R} \right), \tag{3.16}$$

$$I_2 = \frac{1}{4\pi} \int_{PV,S_n} d\boldsymbol{r'}\ \nabla' \left(\frac{1}{R} \right), \tag{3.17}$$

and

$$I_3 = -\frac{k_l^2}{8\pi} \int_{PV,S_n} d\boldsymbol{r'}\ \nabla' R. \tag{3.18}$$

I_1 is calculated using an adaptive integration method or a Gaussian quadrature, whereas I_2 and I_3 are evaluated analytically [41, 22].

Finally, the elements of CFIE matrices can be derived as

$$\left(\overline{\boldsymbol{A}}^{CFIE}\right)_{mn} = \alpha \left(\overline{\boldsymbol{A}}^{EFIE}\right)_{mn} + (1-\alpha) \left(\overline{\boldsymbol{A}}^{MFIE}\right)_{mn}. \tag{3.19}$$

3.4. Computation of the RHS vectors. Elements of the RHS vector for EFIE are obtained by testing the incident electric field in the RHS of (2.1), i.e.,

$$(\boldsymbol{b})_m^{EFIE} = -\frac{i}{k\eta} \int_{S_m} d\boldsymbol{r}\ \boldsymbol{t}_m(\boldsymbol{r}) \cdot \boldsymbol{E}^{inc}(\boldsymbol{r}). \tag{3.20}$$

Similarly, the RHS vector for MFIE can be found using

$$(\boldsymbol{b})_m^{MFIE} = -\int_{S_m} d\boldsymbol{r}\ \boldsymbol{t}_m(\boldsymbol{r}) \cdot \hat{\boldsymbol{n}} \times \boldsymbol{H}^{inc}(\boldsymbol{r}). \tag{3.21}$$

Then the RHS vectors for CFIE can be calculated as the linear combination of (3.20) and (3.21), i.e.,

$$(\boldsymbol{b})_m^{CFIE} = \alpha\, (\boldsymbol{b})_m^{EFIE} + (1-\alpha)\, (\boldsymbol{b})_m^{MFIE}. \tag{3.22}$$

4. The MLFMA. The discretization of EFIE and CFIE with MOM leads to dense linear systems due to the nonlocal nature of the electromagnetic interactions between the basis and testing functions. Surfaces of objects are usually meshed with one-tenth of the wavelength for accuracy. Hence, for high frequencies, where the scatterer or the radiator sizes become large in terms of the wavelength, the system matrix

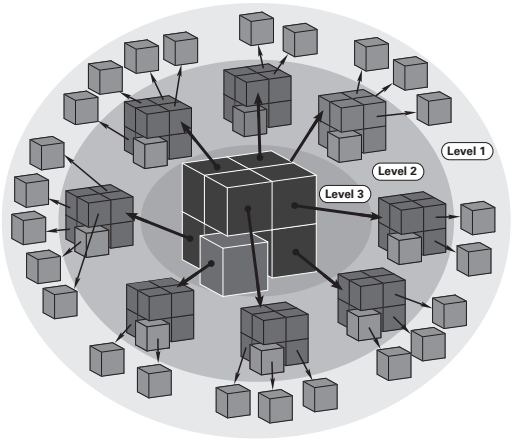

FIG. 4.1. *Illustration of the oct-tree partitioning of the computational domain in MLFMA.*

also becomes large. For solving such matrix systems, direct solution methods become too expensive due to their high computational complexity. Iterative methods may be preferred as a more viable option, provided that the number of iterations remains limited even for large numbers of unknowns. However, iterative methods require matrix-vector multiplications, which have $\mathcal{O}(N^2)$ complexity for $N \times N$ dense matrices. Although lower than the $\mathcal{O}(N^3)$ complexity of direct solvers, $\mathcal{O}(N^2)$ complexity is still prohibitive for large problems. As a result, in addition to effective preconditioners, iterative solutions of real-life CEM problems require acceleration methods for performing fast matrix-vector multiplications with low-complexity. In this context, MLFMA is a method of choice since it renders the solution of large CEM problems possible by reducing the complexity of matrix-vector multiplications to $\mathcal{O}(N \log N)$. The main components of MLFMA are outlined in the following.

4.1. Clustering. In order to compute the interactions between the basis and testing functions in a multilevel scheme, an oct-tree strategy is employed. For this purpose, the whole geometry is placed inside a cube, which is recursively divided into smaller cubes until the smallest cubes contain only a few basis functions, as illustrated in Figure 4.1. If any of the cubes becomes empty during the partitioning, recursion stops there. In any level, pairs of same-size cubes touching at any point are in the near-field zone of each other, and the others are in the far-field zone. In the lowest level (Level 1 in Figure 4.1), interactions between the near-field clusters, including the self-interactions, constitute the near-field matrix, and the remaining far-field interactions constitute the far-field matrix. In the course of an iterative solution, MLFMA decomposes matrix-vector multiplications as

$$\overline{\boldsymbol{A}} \cdot \boldsymbol{x} = \overline{\boldsymbol{A}}^{NF} \cdot \boldsymbol{x} + \overline{\boldsymbol{A}}^{FF} \cdot \boldsymbol{x}. \tag{4.1}$$

In (4.1), $\overline{\boldsymbol{A}}^{NF}$ denotes the near-field matrix, which is calculated directly as described in section 3 and stored in memory to perform the partial matrix-vector multiplication $\overline{\boldsymbol{A}}^{NF} \cdot \boldsymbol{x}$. Examples for $\overline{\boldsymbol{A}}^{NF}$ are depicted in Figure 4.2. Note that these matrices are composed of small blocks, which correspond to the near-field interactions of the lowest-level clusters. However, the matrices do not exhibit any structured sparsity pattern, except for the apparent larger diagonal blocks. Those diagonal blocks are formed from the interactions of the lowest-level clusters that have the same parent

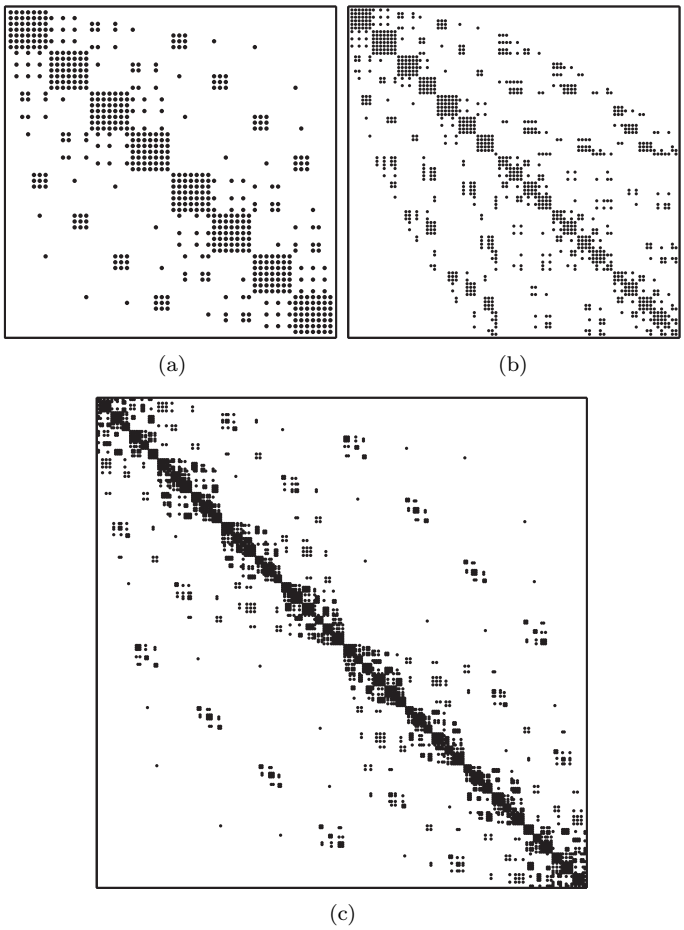

FIG. 4.2. *Sparse near-field matrices for* (a) $n = 930$, (b) $n = 1{,}302$, *and* (c) $n = 3{,}723$.

cluster. $\overline{\boldsymbol{A}}^{FF} \cdot \boldsymbol{x}$ denotes the multiplication with far-field interactions, which will be detailed in section 4.3. To achieve $\mathcal{O}(N \log N)$ complexity, this stage is performed approximately but with controllable error, i.e., with the desired level of accuracy.

4.2. Factorization of the Green's function. MLFMA is proposed as a multilevel extension of the single-level FMM [23, 9], and the factorization of the Green's function is at the core of FMM.

Consider two far-zone clusters that are defined with the reference points C' and C. For the interactions between the basis functions that are clustered around C' and testing functions that are clustered around C, the scalar Green's function can be factorized as [35]

$$g(\boldsymbol{r}, \boldsymbol{r'}) = \frac{e^{ik|\boldsymbol{r}-\boldsymbol{r'}|}}{4\pi|\boldsymbol{r}-\boldsymbol{r'}|} = \frac{e^{ik|\boldsymbol{D}+\boldsymbol{d}|}}{4\pi|\boldsymbol{D}+\boldsymbol{d}|} \approx \frac{1}{4\pi} \int d^2\hat{\boldsymbol{k}}\; e^{i\hat{\boldsymbol{k}}\cdot\boldsymbol{d}} \alpha_T(k, D, \hat{\boldsymbol{D}} \cdot \hat{\boldsymbol{k}}), \tag{4.2}$$

where $D = |\boldsymbol{D}|$ represents the distance between C' and C. The integration in (4.2) is performed on the unit sphere, and $\hat{\boldsymbol{k}}$ is the unit vector normal to the unit sphere.

The translation function

$$\alpha_T(k, D, \hat{\boldsymbol{D}} \cdot \hat{\boldsymbol{k}}) = \sum_{t=0}^{T} i^t (2t+1) h_t^{(1)}(kD) P_t(\hat{\boldsymbol{D}} \cdot \hat{\boldsymbol{k}}) \tag{4.3}$$

involves the spherical Hankel function of the first kind $h_t^{(1)}$ and the Legendre polynomial P_t. The translation function defined in (4.3) can be used to evaluate the group interactions between the basis and testing functions clustered around C' and C, instead of calculating the interactions separately.

By diagonalizing [10] the scalar Green's function as in (4.2) and (4.3), single-level interactions can be derived as

$$\left(\overline{\boldsymbol{A}}\right)_{mn} = \left(\frac{ik}{4\pi}\right)^2 \int d^2\hat{\boldsymbol{k}}\, \overline{\boldsymbol{F}}_{Cm}^{rec}(\hat{\boldsymbol{k}}) \cdot \alpha_T(k, D, \hat{\boldsymbol{D}} \cdot \hat{\boldsymbol{k}})\, \overline{\boldsymbol{F}}_{C'n}^{rad}(\hat{\boldsymbol{k}}), \tag{4.4}$$

where $\overline{\boldsymbol{F}}_{Cm}^{rec}$ represents the receiving pattern of the mth testing function with respect to the reference point C and $\overline{\boldsymbol{F}}_{C'n}^{rad}$ represents the radiation pattern of the nth basis function with respect to the reference point C'.

In any MLFMA level l, radiation and receiving patterns are defined and sampled at $\mathcal{O}(T_l^2)$ angular points, where T_l is the truncation number for the series in (4.3). Since we set the minimum cluster size at the lowest level as 0.25λ, the cluster size at level l is $a_l = 2^{l-3}\lambda$. For a cluster of size a_l, the truncation number is determined by using the excess bandwidth formula [29] for the worst-case scenario and the one-box buffer scheme [27], i.e.,

$$T_l \approx 1.73ka + 2.16(d_0)^{2/3}(ka_l)^{1/3}, \tag{4.5}$$

where d_0 is the number of accurate digits desired.

4.3. Far-field interactions. In MLFMA, far-field interactions are calculated in a multilevel scheme and in a group-by-group manner. For this purpose, the aggregation, translation, and disaggregation stages are performed in each matrix-vector multiplication. These stages are described below.

- *Aggregation.* Radiated fields of clusters are calculated from the bottom of the tree structure to the highest level. At the lowest level, radiation patterns of basis functions are multiplied with the elements of the input vector provided by the iterative solver. Then the radiated field of a cluster is determined by combining the radiation patterns inside the cluster. At higher levels, the radiated field of a cluster is obtained by combining the radiated fields of the clusters in the lower levels. Between two consecutive levels, interpolations are employed to match the different sampling rates of the fields using a local interpolation method [14, 16].
- *Translation.* For each pair of far-field clusters whose parents are in the near-field zone of each other, the cluster-to-cluster interaction is computed via a translation. Note that the sizes of the cubic clusters are identical in each level. Hence, the number of translation operators is reduced to $\mathcal{O}(1)$ using the symmetry. For those clusters whose parents are in the far-field zone of each other, the cluster-to-cluster interaction is performed in a higher-level translation.
- *Disaggregation.* Total incoming fields at the cluster centers are calculated from the top of the tree structure to the lowest level. The total incoming

field for a cluster is obtained by combining incoming fields due to translations and the incoming field from its parent cluster if it exists. Incoming fields to the center of a cluster are shifted to the centers of the clusters in the lower levels by using transpose interpolations, or anterpolations [5]. Finally, in the lowest level, incoming fields are received by the testing functions via angular integrations.

5. Preconditioning of integral-equation methods. Developing effective preconditioners for real-life CEM problems is crucial for a number of reasons. Among the surface integral equations, the most accurate results can be obtained via EFIE, but EFIE produces ill-conditioned systems. In addition, EFIE is the only choice for problems involving open surfaces. For closed-surface problems, CFIE can be used, and it produces better-conditioned systems. However, for high-frequency simulations of complex targets, e.g., helicopters or stealth airborne targets, the number of iterations can still be high [32]. Furthermore, at some frequencies where a physical resonance occurs, further increases in numbers of iterations are observed [26].

5.1. Near-field versus full-matrix preconditioners. Since only the interactions corresponding to the lowest-level near-field clusters are kept in memory, it is common practice to construct preconditioners from $\overline{\boldsymbol{A}}^{NF}$, assuming that it is a good approximation to $\overline{\boldsymbol{A}}$. However, since the size of the lowest-level clusters is kept fixed in MLFMA, the number of nonzero elements in a row of $\overline{\boldsymbol{A}}^{NF}$ also remains constant. Therefore, $\overline{\boldsymbol{A}}^{NF}$ becomes increasingly sparser as the problem size grows. As a result, it has been shown that preconditioners that make use of the full $\overline{\boldsymbol{A}}$ matrix, as in some nested-solver schemes [1], are usually stronger than preconditioners that depend on only near-field interactions [7, 21].

Nonetheless, for matrices obtained from the discretizations of surface integral equations, magnitudes of matrix elements change with physical proximity, as a general trend. Therefore, the available near-field matrix $\overline{\boldsymbol{A}}^{NF}$ is likely to preserve the most relevant contributions of the dense system matrix. As section 6 will reveal, the proposed preconditioner renders solutions of large EFIE problems possible with modest iteration counts by effectively using all information provided by the near-field matrix. The results will also reveal that the scaling of iteration counts with respect to increasing problem sizes is remarkably favorable, e.g., iteration counts increase less than threefold, even when problem sizes increase thirty-six-fold in some cases. Furthermore, once a fixed preconditioner, such as SAI, is constructed, the proposed scheme has no extra costs in terms of setup time and memory. On the other hand, preconditioners that make use of the full matrix require less-accurate versions of MLFMA, which can be obtained using extra setup time and significant amounts of memory.

5.2. SAI preconditioner. By using the near-field matrix as a preconditioning matrix, preconditioners proposed for the iterative solution of sparse linear systems can be adapted to integral-equation methods. Typical members of this class are the incomplete factorization methods, which are based on eliminating some of the entries during the LU factorization [3]. After decomposing the near-field matrix in the form of

$$\overline{\boldsymbol{A}}^{NF} \approx \overline{\boldsymbol{L}} \cdot \overline{\boldsymbol{U}}, \tag{5.1}$$

preconditioning is performed in each step by solving

$$\overline{\boldsymbol{L}} \cdot \overline{\boldsymbol{U}} \cdot \boldsymbol{v} = \boldsymbol{w}, \tag{5.2}$$

where $\overline{\boldsymbol{L}}$ and $\overline{\boldsymbol{U}}$ are the incomplete factors. On the other hand, an SAI matrix $\overline{\boldsymbol{M}}$ directly approximates the inverse of the matrix $\overline{\boldsymbol{A}}$, and the preconditioner is applied simply with the sparse matrix-vector multiplication $\boldsymbol{v} = \overline{\boldsymbol{M}} \cdot \boldsymbol{w}$. Backward and forward substitutions required in the incomplete factorization methods are inherently sequential; hence for parallel applications, approximate-inverse preconditioners are preferred.

In a broad sense, there are three types of SAI preconditioners, i.e., factorized approximate inverses, inverse ILU techniques, and SAIs that depend on Frobenius norm minimization [4]. Among them, the one based on Frobenius norm minimization has been successfully used in CEM problems [7, 30, 32]. In this method, the approximate inverse of the near-field matrix is computed by minimizing

$$\left\| \overline{\boldsymbol{I}} - \overline{\boldsymbol{M}} \cdot \overline{\boldsymbol{A}}^{NF} \right\|_F . \tag{5.3}$$

The approximation arises from forcing $\overline{\boldsymbol{M}}$ to be sparse. Minimization can be performed independently for each row by using the identity

$$\left\| \overline{\boldsymbol{I}} - \overline{\boldsymbol{M}} \cdot \overline{\boldsymbol{A}}^{NF} \right\|_F^2 = \sum_{i=1}^{n} \left\| \boldsymbol{e}_i - \boldsymbol{m}_i \cdot \overline{\boldsymbol{A}}^{NF} \right\|_2^2 , \tag{5.4}$$

where $\boldsymbol{e}_i$ is the ith unit row vector and $\boldsymbol{m}_i$ is the ith row of the preconditioner. The nonzero pattern of $\overline{\boldsymbol{M}}$ is fixed in advance. Usually, the pattern of $\overline{\boldsymbol{A}}^{NF}$ is preferred, but filtered patterns may also be adequate to reduce memory costs for some specific cases where the near-field matrices require substantial memory [26].

5.3. Iterative near-field preconditioner. It is known that SAI is not as successful as ILU with the same amount of memory [4]. We confirm this assertion by comparing SAI with the exact solution of the near-field matrix, which we name NF-LU. Though ILUT produces iteration counts very close to those of NF-LU [31], SAI deviates from this optimum behavior as the number of unknowns increases. For a remedy, increasing the density of the preconditioner is undesirable because of possible high setup time and memory considerations.

On the other hand, an iterative solution of the near-field matrix can be used as a preconditioner, provided that the original system is solved using a flexible solver [37]. Since SAI is a good approximation to the inverse of the near-field matrix, the iterative solution of the near-field system can be accelerated using SAI as a preconditioner. This approach produces a nesting of the iterative solvers. For the outer solver that solves the original system, we use the flexible GMRES (FGMRES) method, which allows the preconditioner to change from iteration to iteration [37]. The preconditioner of this solver is another preconditioned Krylov subspace solver, which we call the inner solver. We solve the sparse near-field system in the inner solver using SAI as the fixed preconditioner. We illustrate this nested inner-outer preconditioning scheme in Figure 5.1.

Since the inner solver is used for preconditioning purposes, a rough solution is adequate. We use GMRES as the inner solver since it provides a fast drop of the residual norm in early iterations.

The proposed scheme, which we name the iterative near-field (INF) preconditioner, yields a forward-type preconditioner, as the ILU preconditioner is. The difference is that, in ILU preconditioning, the preconditioner approximates the near-field matrix in factorized form, i.e., $\overline{\boldsymbol{M}} = \overline{\boldsymbol{L}} \cdot \overline{\boldsymbol{U}} \approx \overline{\boldsymbol{A}}^{NF}$, but the system $\overline{\boldsymbol{M}} \cdot \boldsymbol{v} = \boldsymbol{w}$ is solved exactly by using backward and forward solves for a given vector $\boldsymbol{w}$. On the

Outer solver:	FGMRES; solve $\overline{\boldsymbol{A}} \cdot \boldsymbol{x} = \boldsymbol{b}$
	Matrix-vector product: MLFMA
Inner solver (preconditioner):	GMRES; solve $\overline{\boldsymbol{A}}^{NF} \cdot \boldsymbol{v} = \boldsymbol{w}$
	Matrix-vector product: Sparse mat-vec
	Fixed preconditioner: SAI

FIG. 5.1. *Nested solvers for iterative near-field preconditioning.*

other hand, for INF, the preconditioner is the exact near-field matrix, i.e., $\overline{\boldsymbol{M}} = \overline{\boldsymbol{A}}^{NF}$, but we approximately solve the system $\overline{\boldsymbol{M}} \cdot \boldsymbol{v} = \boldsymbol{w}$ with an iterative method.

6. Results. In this section, we compare the performances of the SAI and INF preconditioners since the SAI preconditioner has been widely used and proven to be successful in parallel implementations of integral-equation methods [2, 30, 6, 32]. Furthermore, when the near-field matrix pattern is selected as the nonzero pattern of the approximate inverse, the setup time of the SAI preconditioner can be lowered using the block structure, as shown in [7, 32]. Regarding the stopping criteria of the inner solver for the INF preconditioner, we conclude that the one-order residual drop provides a successful preconditioner that can be attained in a few iterations. Hence, we set the stopping criteria of the inner solver as a one-order residual drop from the initial residual norm or a maximum of five iterations, whichever is satisfied first.

For small problems, we can evaluate the quality of the SAI and INF preconditioners by comparing them with a preconditioner obtained from the exact factorization of the near-field matrix. This preconditioner, which we call NF-LU, can be used only as a benchmark due to its excessive memory and setup costs. Nonetheless, it is useful for evaluation purposes, since its iteration count is expected to be the minimum that can be achieved with a preconditioner constructed from the near-field matrix. Then we can evaluate other preconditioners on the basis of how close their iteration counts are to those of NF-LU.

In Table 6.1, we present the solutions of three geometries with various preconditioners, i.e., the diagonal preconditioner (DP), SAI, INF, NF-LU, and the no-preconditioner case (No PC). Computations are performed on a 16-core parallel cluster constructed with eight dual-core AMD Opteron 870 processors in a symmetric multiprocessing configuration. The geometries are depicted in Figure 6.1. We choose geometries with open surfaces, since closed-surface geometries can be solved more easily using CFIE. Mesh size is chosen as one-tenth of the wavelength at the frequency of operation. Due to its robustness, we use GMRES (FGMRES for INF) with no-restart as the iterative solver. We set the the initial guess as a vector of zeros and the stopping criterion as either a six–order of magnitude relative decay from the initial residual or a maximum of 1,000 iterations. In our MLFMA implementation, the size of the smallest clusters is fixed to 0.25 wavelength and the number of accurate digits to three.

The results presented in Table 6.1 show that the SAI preconditioner succeeds in accelerating the convergence of these relatively small problems since their solutions without a preconditioner or with DP require either several hundreds of or more than 1,000 iterations. On the other hand, the iteration counts are not close to those of NF-LU. This observation can be interpreted as that there is more room for improvement between an approximate inverse generated with the SAI preconditioner and the exact inverse computed with NF-LU for benchmarking purposes. One can actually

TABLE 6.1
Experimental results for comparing the SAI and INF preconditioners to NF-LU.

Geometry	N	No PC		DP		SAI		INF		NF-LU
		Iter	Time	Iter	Time	Iter	Time	Iter	Time	Iter
Patch	12,249	447	106	432	103	44	12	29	9	26
	137,792	894	3,241	851	3,087	91	336	59	253	53
Half Sphere	9,911	514	178	485	438	60	24	40	17	38
	116,596	-	3,257	-	3,259	156	510	103	383	93
Reflector Antenna	12,142	564	453	545	458	44	22	28	13	27
	105,570	-	4,285	-	4,288	80	344	51	236	49

Notes: "Iter" denotes the number of iterations and "Time" denotes the solution times.
A dash "-" indicates that convergence is not attained in 1,000 iterations.

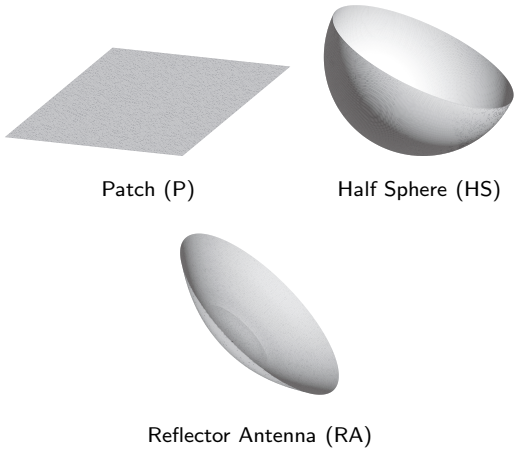

FIG. 6.1. *Open-surface geometries used for comparing the preconditioners.*

increase the density of the approximate inverses using two different tree structures for MLFMA and for the construction of the SAI preconditioner, as detailed in [7], but this comes at the cost of extra memory, which is a potential source of problems for large CEM computations. With the INF preconditioner, however, we achieve iteration counts that are very close to those of NF-LU. This means that the INF preconditioner makes good use of the available sparse near-field matrix and produces nearly optimal approximations for the inverse. In addition, these approximations are achieved in at most five iterations; hence the solution times are also decreased significantly.

To further assess the performance of the INF preconditioner, we solve larger instances of the problems in Figure 6.1 with increasing frequencies, as shown in Table 6.2. The solutions of these problems are carried out on 32 cores of an eight-node cluster interconnected with an Infiniband network. Each node of the cluster has two Intel Xeon 5345 quad-core processors and 32 GB of RAM. We note that none of the problems in Table 6.2 can be solved without an effective preconditioner even if the no-restart GMRES solver is used.

Iteration counts and timings pertaining to the solutions of the problems listed in Table 6.2 for the SAI and INF preconditioners are presented in Table 6.3. These results indicate that the proposed INF preconditioner consistently achieves better performance than the SAI preconditioner in all cases. The INF preconditioner decreases

TABLE 6.2
Quantitative features of the open-surface geometries used for the numerical experiments.

Geometry	Frequency (GHz)	Size (λ)	MLFMA levels	N
P1	32	32	8	344,000
P2	64	64	9	1,377,280
P3	96	96	10	3,062,400
P4	128	128	11	5,511,680
HS1	32	64	9	408,064
HS2	64	96	10	1,633,280
HS3	96	192	10	3,838,496
HS4	128	256	11	6,535,168
RA1	8	27	8	187,144
RA2	16	53	9	748,024
RA3	32	107	10	2,991,067
RA4	48	160	11	6,849,398

Notes: "Size" denotes the edge length for the patch and the diameter for the sphere. λ denotes the wavelength at the frequency of operation.

TABLE 6.3
Experimental results for comparing the SAI and INF preconditioners.

Geometry	SAI setup	SAI		INF		
		Iter	Time	Inner iter	Outer iter	Time
P1	10	109	174	217	73	132
P2	48	157	1,147	316	106	812
P3	132	194	6,225	391	131	4,393
P4	308	234	27,902	478	160	19,620
HS1	20	221	1,424	480	160	999
HS2	92	351	10,046	780	260	7,258
HS3	350	480	23,458	1,101	367	18,374
HS4	839	546	66,778	1,218	406	51,285
RA1	9	93	204	184	62	136
RA2	37	139	1,266	272	95	832
RA3	201	200	7,276	408	138	5,138
RA4	671	252	31,784	509	172	22,404

Notes: "SAI setup"denotes the construction time of SAI (in seconds) and applies to both SAI and INF. "Time" denotes the solution times, given in seconds. "Inner iter" and "Outer iter" denote the total number of inner and outer iterations, respectively.

the solution times of the patch and reflector antenna problems by about 30% and those of the half-sphere problem by about 25%, with respect to the SAI preconditioner.

In each iteration, GMRES stores the preconditioned residual vector [37]; hence its memory cost can be significant for large problems when the number of iterations is high. In Table 6.4, we present the parallel memory costs (per process) of GMRES for solutions with SAI and INF preconditioners. We also present the memory consumptions of MLFMA and the SAI setup. Since the sparsity pattern of SAI is the same as that of the near-field matrix, we do not need to store the indexing arrays for SAI [32]. As a result, the amount of memory required by SAI is much less than that of MLFMA. On the other hand, memory requirements of GMRES are significant, and they are even higher than those of the SAI setup. INF reduces the iteration counts with respect to the SAI preconditioner, but GMRES memory for INF is larger than

TABLE 6.4
Memory costs (in MB) of MLFMA, SAI/INF setup, and GMRES solutions.

Geometry	MLFMA	SAI/INF setup	GMRES	
			SAI	INF
P1	78	16	9	12
P2	261	64	52	70
P3	430	139	142	191
P4	2,955	256	307	421
HS1	201	17	22	31
HS2	788	69	137	202
HS3	1,769	169	439	672
HS4	3,145	277	851	1,265
RA1	87	8	4	6
RA2	327	33	25	34
RA3	1,274	133	143	197
RA4	3,114	313	407	556

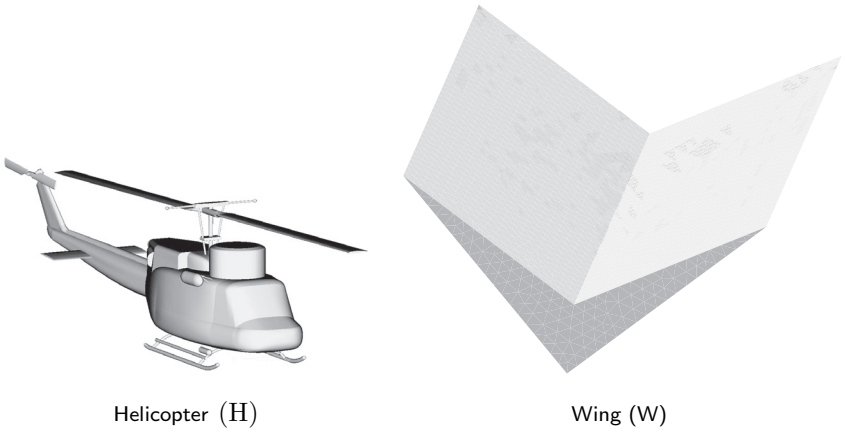

FIG. 6.2. *Closed-surface geometries formulated with CFIE.*

that of SAI, since for INF we use the flexible version of GMRES, whose memory cost is twice that of usual GMRES. Nonetheless, we note that the memory consumption of GMRES is much less than that of MLFMA.

We investigate the performance of the INF preconditioner on two closed-surface problems formulated with CFIE. Even though CFIE is expected to produce better-conditioned systems compared to the EFIE formulation of open geometries, the two closed-surface problems are selected as particularly difficult real-life problems. These problems involve a wing geometry (W) and a helicopter (H), as illustrated in Figure 6.2. The wing geometry has sharp edges and corners. The helicopter geometry has a closed surface, but with very thin features and complicated surfaces, causing the deterioration of its condition numbers. Quantitative features of various numerical experiments are listed in Table 6.5. Both the wing (W) and the helicopter (H) problems are discretized with very large numbers of unknowns: 7.5 million and 13 million, respectively. Furthermore, the surface of the real-life helicopter (H) geometry is triangulated with three different mesh types, and each mesh type is created with three different mesh sizes, hence obtaining 9 different problems. For example, $H3_1$ in Table 6.5 denotes the third mesh size for the first mesh type. This is a very realistic

TABLE 6.5
Quantitative features of the closed-surface geometries used for the numerical experiments.

Geometry	Frequency (GHz)	Size (λ)	MLFMA levels	N
W1	4	13	7	117,945
W2	8	27	8	471,780
W3	16	53	9	1,887,120
W4	32	107	10	7,548,480
$H1_1$	1.3	74	10	556,515
$H2_1$	2.6	147	11	2,226,060
$H3_1$	5.2	295	12	8,904,240
$H1_2$	1.4	79	10	644,133
$H2_2$	2.8	159	11	2,576,532
$H3_2$	5.6	317	12	10,306,128
$H1_3$	1.6	91	10	817,260
$H2_3$	3.2	181	11	3,269,040
$H3_3$	6.4	363	12	13,076,160

Notes: "Size" denotes the largest dimension, i.e., edge length of the smallest cube enclosing the geometry. λ denotes the wavelength at the frequency of operation.

TABLE 6.6
Experimental results for comparing the INF preconditioner with DP, BDP, and the SAI preconditioner for closed-surface problems.

Geometry	DP		BDP		SAI	SAI		INF		
	Iter	Time	Iter	Time	setup	Iter	Time	Inner	Iter	Time
W1	100	61	60	37	12	42	34	60	31	22
W2	127	300	78	186	33	57	150	78	40	111
W3	166	1,667	98	985	111	74	832	103	53	617
W4	211	8,951	131	5,559	576	96	4,139	212	65	3,166
$H1_1$	170	2,722	115	1,848	54	75	1,249	202	55	960
$H2_1$	170	12,581	115	8,490	172	92	7,026	222	74	5,771
$H3_1$	195	65,151	134	44,804	644	112	38,164	273	91	31,293
$H1_2$	169	2,996	110	1,871	62	77	1,374	197	57	1,045
$H2_2$	170	12,892	114	8,655	215	94	7,454	234	78	6,325
$H3_2$	205	74,821	136	48,416	856	117	42,836	283	94	35,072
$H1_3$	167	3,032	150	2,730	70	79	1,513	197	59	1,168
$H2_3$	177	14,209	160	12,886	267	98	8,270	240	80	6,915
$H3_3$	205	77,240	187	70,549	1,054	127	49,344	297	99	39,268

Notes: "SAI setup" denotes the construction time of SAI (in seconds) and applies to both SAI and INF. "Time" denotes the solution times, given in seconds. "Iter" denotes the number of iterations, and "Inner" denotes the total number of iterations of the inner solver.

approach, since different mesh generators and different users of mesh generators produce different types of meshes, which, in turn, influence the condition of the resulting matrix equations. We will demonstrate the effectiveness of the INF preconditioner on these difficult real-life problems.

For the closed-surface problems, in addition to DP, SAI, and INF, we consider also the BDP, which is commonly used with the CFIE formulation. BDP is obtained by exactly solving the diagonal blocks that represent the self-interactions of the lowest-level clusters of the MLFMA tree structure (Figure 4.1). Iteration counts and timings of the solutions are compared in Table 6.6. With the INF preconditioner, we observe

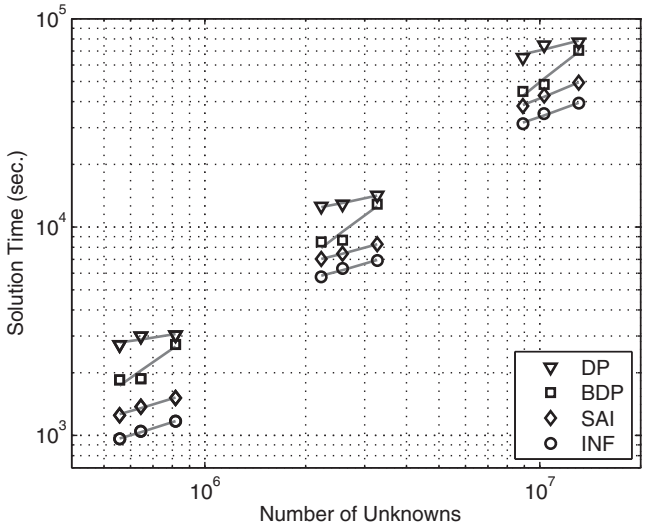

FIG. 6.3. *Total solution times of the helicopter problem. Least-squares best-fit lines are also shown. The three groups correspond to the three MLFMA levels. The INF preconditioner consistently provides faster solutions than the other preconditioners.*

a significant decrease in the solution time. For the wing geometry, the gain is about 40% with respect to BDP and 25% to 35% with respect to the SAI preconditioner. For the real-life helicopter problem, which has thin and complicated surfaces, the gain is about 27% to 57% with respect to BDP, and 16% to 25% with respect to the SAI preconditioner.

We further analyze helicopter solutions in Figure 6.3, where we plot the total solution times, including the setup and solution times of the preconditioner. For all instances of problem sizes and mesh types, the INF preconditioner consistently provides faster solutions than the other preconditioners. Figure 6.3 shows that all solution times obey the $\mathcal{O}(N \log N)$ complexity of MLFMA, in general. As the problem sizes grow and MLFMA levels increase, it is well known that the solution times experience discrete jumps [18], without violating the general $\mathcal{O}(N \log N)$ complexity. For this reason, we plot the solution times in the three groups, corresponding to the three MLFMA levels, i.e., 10, 11, and 12. In each group, the solution times with the INF preconditioner are significantly lower than those with the other preconditioners, especially considering that the vertical axis in Figure 6.3 is scaled logarithmically.

Finally, in Table 6.7, we present the parallel memory costs (per process) for CFIE solutions. We include the group that contains the largest helicopter problem. Closed-surface problems can be solved with CFIE in fewer iterations, compared to open-surface problems solved with EFIE. Therefore, memory required by the GMRES solver is significantly less than those presented in Table 6.4. Even though the memory requirement of FGMRES employed by INF is higher than that of GMRES, the memory cost of INF is not significant compared to that of MLFMA. We also note that the memory costs of the SAI setup and GMRES are much less than that of MLFMA.

TABLE 6.7
Memory costs (in MB) of MLFMA, SAI/INF setup, and GMRES solutions.

Geometry	MLFMA	SAI/INF setup	GMRES			
			DP	BDP	SAI	INF
W1	68	7	3	2	1	2
W2	232	26	14	9	6	9
W3	774	97	75	44	33	48
W4	2,360	371	380	236	173	234
$H1_3$	437	46	33	29	15	23
$H2_3$	1,831	167	138	125	76	125
$H3_3$	7,431	637	639	583	396	617

7. Conclusion. For the iterative solution of EFIE via MLFMA, designing preconditioners that effectively use the information provided by the sparse near-field matrix is crucial for fast convergence. Even though the CFIE formulation yields better-conditioned linear systems than EFIE, its use is limited to closed-surface problems. Furthermore, real-life problems usually involve thin and complex parts, and this causes an increase in the iteration counts required for convergence, even with CFIE. Hence, iterative solutions of CFIE also benefit from preconditioning. ILU [31] and SAI [32] preconditioners are designed for this purpose. ILU preconditioners are not suitable for scalable parallel implementations. SAI preconditioners accelerate the iterative convergence to some extent, but they have limited success in taking full advantage of the available sparse near-field matrix, as demonstrated by the comparisons with the benchmark LU solutions (NF-LU) in Table 6.1. To increase their effectiveness, one can increase the density of the approximate inverses beyond that of the near-field matrix, but this is not the best solution because of the memory considerations. Moreover, the benefit obtained even with this costly solution is limited, as shown in [6].

In this work, we propose an alternative way to increase the efficiency using flexible solvers. In this scheme, the near-field matrix is iteratively solved and used as a preconditioner in addition to a fixed preconditioner, such as an SAI preconditioner, which is used to accelerate the inner iterative solver. This approach has the following advantages:

- By using the available SAI as the preconditioner of the inner system, only a few iterations suffice to achieve a strong preconditioner, and the iteration counts of the outer solver become very close to those obtained from the benchmark exact solution of the near-field system. Hence, the cost of applying the preconditioner is lowered, and the overall solution times are significantly decreased.
- The proposed INF preconditioner is demonstrated to provide faster (i.e., shorter CPU times and fewer iterations) and scalable solutions for problems involving as many as 13 million unknowns. The advantage of the INF preconditioner over the other near-field preconditioners is consistent and does not vanish as the problem size grows.
- The proposed preconditioner's parallel scalability is very good because the application of the preconditioner consists merely of repeated sparse matrix-vector multiplications, which are highly parallelizable.
- The only cost of the proposed scheme is the extra storage of the preconditioned residual vectors of FGMRES in memory, because of the variable

preconditioning [37]. However, since the iteration counts are reduced, the required FGMRES memory is not significant, especially compared to the MLFMA memory.

REFERENCES

[1] K. Abe and S.-L. Zhang, *A variable preconditioning using the SOR method for GCR-like methods*, Int. J. Numer. Anal. Model., 2 (2005), pp. 147–161.

[2] G. Alléon, M. Benzi, and L. Giraud, *Sparse approximate inverse preconditioning for dense linear systems arising in computational electromagnetics*, Numer. Algorithms, 16 (1997), pp. 1–15.

[3] M. Benzi, *Preconditioning techniques for large linear systems: A survey*, J. Comput. Phys., 182 (2002), pp. 418–477.

[4] M. Benzi and M. Tuma, *A comparative study of sparse approximate inverse preconditioners*, Appl. Numer. Math., 30 (1999), pp. 305–340.

[5] A. Brandt, *Multilevel computations of integral transforms and particle interactions with oscillatory kernels*, Comput. Phys. Comm., 65 (1991), pp. 24–38.

[6] B. Carpentieri, I. S. Duff, and L. Giraud, *Sparse pattern selection strategies for robust Frobenius-norm minimization preconditioners in electromagnetism*, Numer. Linear Algebra Appl., 7 (2000), pp. 667–685.

[7] B. Carpentieri, I. S. Duff, L. Giraud, and G. Sylvand, *Combining fast multipole techniques and an approximate inverse preconditioner for large electromagnetism calculations*, SIAM J. Sci. Comput., 27 (2005), pp. 774–792.

[8] K. Chen, *Matrix Preconditioning Techniques and Applications*, Cambridge Monogr. Appl. Comput. Math., 19, Cambridge University Press, Cambridge, UK, 2005.

[9] W.-C. Chew, J.-M. Jin, E. Michielssen, and J. Song, eds., *Fast and Efficient Algorithms in Computational Electromagnetics*, Artech House, Norwood, MA, 2001.

[10] R. Coifman, V. Rokhlin, and S. Wandzura, *The fast multipole method for the wave equation: A pedestrian prescription*, IEEE Antennas Propag. Mag., 35 (1993), pp. 7–12.

[11] D. A. Dunavant, *High degree efficient symmetrical Gaussian quadrature rules for the triangle*, Internat. J. Numer. Methods Engrg., 21 (1985), pp. 1129–1148.

[12] Ö. Ergül, *Fast Multipole Method for the Solution of Electromagnetic Scattering Problems*, Master's thesis, Bilkent University, Ankara, Turkey, 2003.

[13] Ö. Ergül and L. Gürel, *Improved testing of the magnetic-field integral equation*, IEEE Microw. Wireless Comp. Lett., 15 (2005), pp. 615–617.

[14] Ö. Ergül and L. Gürel, *Enhancing the accuracy of the interpolations and anterpolations in MLFMA*, IEEE Antennas Wireless Propagat. Lett., 5 (2006), pp. 467–470.

[15] Ö. Ergül and L. Gürel, *Improving the accuracy of the magnetic-field integral equation with the linear-linear basis functions*, Radio Sci., 41 (2006), RS4004, doi:10.1029/2005RS003307.

[16] Ö. Ergül and L. Gürel, *Optimal interpolation of translation operator in multilevel fast multipole algorithm*, IEEE Trans. Antennas and Propagation, 54 (2006), pp. 3822–3826.

[17] Ö. Ergül and L. Gürel, *The use of curl-conforming basis functions for the magnetic-field integral equation*, IEEE Trans. Antennas and Propagation, 54 (2006), pp. 1917–1926.

[18] Ö. Ergül and L. Gürel, *Efficient parallelization of the multilevel fast multipole algorithm for the solution of large-scale scattering problems*, IEEE Trans. Antennas and Propagation, 56 (2008), pp. 2335–2345.

[19] Ö. Ergül and L. Gürel, *Discretization error due to the identity operator in surface integral equations*, Comput. Phys. Comm., 180 (2009), pp. 1746–1752.

[20] Ö. Ergül and L. Gürel, *A hierarchical partitioning strategy for an efficient parallelization of the multilevel fast multipole algorithm*, IEEE Trans. Antennas and Propagation, 57 (2009), pp. 1740–1750.

[21] Ö. Ergül, T. Malas, and L. Gürel, *Solutions of Large-scale Electromagnetics Problems using an Iterative Inner-outer Scheme with Ordinary and Approximate Multilevel Fast Multipole algorithms*, Technical report, Bilkent University, Ankara, Turkey, 2009.

[22] R. D. Graglia, *On the numerical integration of the linear shape functions times the 3-D Green's function or its gradient on a plane triangle*, IEEE Trans. Antennas and Propagation, 41 (1993), pp. 1448–1455.

[23] L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, J. Comput. Phys., 73 (1987), pp. 325–348.

[24] L. Gürel and Ö. Ergül, *Comparisons of FMM implementations employing different formulations and iterative solvers*, in IEEE Antennas Propagat. Society Int. Symp. 1, 2003, pp. 19–22.
[25] L. Gürel and Ö. Ergül, *Singularity of the magnetic-field integral equation and its extraction*, IEEE Antennas Wireless Propagat. Lett., 4 (2005), pp. 229–232.
[26] L. Gürel, Ö. Ergül, A. Ünal, and T. Malas, *Fast and accurate analysis of large metamaterial structures using the multilevel fast multipole algorithm*, Prog. Electromagn. Res. (PIER), 95 (2009), pp. 179–198.
[27] M. Larkin Hastriter, S. Ohnuki, and W.-C. Chew, *Error control of the translation operator in 3D MLFMA*, Microw. Opt. Technol. Lett., 37 (2003), pp. 184–188.
[28] R. E. Hodges and Y. Rahmat-Samii, *The evaluation of MFIE integrals with the use of vector triangle basis functions*, Microw. Opt. Technol. Lett., 14 (1997), pp. 9–14.
[29] S. Koc, J. Song, and W.-C. Chew, *Error analysis for the numerical evaluation of the diagonal forms of the scalar spherical addition theorem*, SIAM J. Numer. Anal., 36 (1999), pp. 906–921.
[30] J. Lee, J. Zhang, and C.-C. Lu, *Sparse inverse preconditioning of multilevel fast multipole algorithm for hybrid integral equations in electromagnetics*, IEEE Trans. Antennas and Propagation, 52 (2004), pp. 2277–2287.
[31] T. Malas and L. Gürel, *Incomplete LU preconditioning with the multilevel fast multipole algorithm for electromagnetic scattering*, SIAM J. Sci. Comput., 29 (2007), pp. 1476–1494.
[32] T. Malas and L. Gürel, *Accelerating the multilevel fast multipole algorithm with the sparse-approximate-inverse (SAI) preconditioning*, SIAM J. Sci. Comput., 31 (2009), pp. 1968–1984.
[33] S. M. Rao and D. R. Wilton, *E-field, H-field, and combined-field solution for arbitrarily shaped three-dimensional dielectric bodies*, Electromag., 10 (1990), pp. 407–421.
[34] S. M. Rao, D. R. Wilton, and A. W. Glisson, *Electromagnetic scattering by surfaces of arbitrary shape*, IEEE Trans. Antennas and Propagation, AP-30 (1982), pp. 409–418.
[35] V. Rokhlin, *Rapid solution of integral equations of scattering theory in two dimensions*, J. Comput. Phys., 86 (1990), pp. 414–439.
[36] Y. Saad, *ILUT: A dual threshold incomplete LU factorization*, Numer. Linear Algebra Appl., 1 (1994), pp. 387–402.
[37] Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd ed., SIAM, Philadelphia, 2003.
[38] X. Qing Sheng, J.-M. Jin, J. Song, W.-C. Chew, and C.-C. Lu, *Solution of combined-field integral equation using multilevel fast multipole algorithm for scattering by homogeneous bodies*, IEEE Trans. Antennas and Propagation, 46 (1998), pp. 1718–1726.
[39] L. N. Trefethen, *Computation of pseudospectra*, Acta Numer., 8 (1999), pp. 247–295.
[40] D. R. Wilton and J. E. Wheeler III, *Comparison of convergence rates of the conjugate gradient method applied to various integral equation formulations*, Prog. Electromagn. Res. (PIER), 5 (1991), pp. 131–158.
[41] P. Ylä-Oijala and M. Taskinen, *Calculation of CFIE impedance matrix elements with RWG and $\hat{\boldsymbol{n}}\times$RWG functions*, IEEE Trans. Antennas and Propagation, 51 (2003), pp. 1837–1846.
[42] P. Ylä-Oijala, M. Taskinen, and S. Järvenpää, *Surface integral equation formulations for solving electromagnetic scattering problems with iterative methods*, Radio Sci., 40 (2005), RS6002, doi:10.1029/2004RS003169.
[43] P. Ylä-Oijala, M. Taskinen, and S. Järvenpää, *Analysis of surface integral equations in electromagnetic scattering and radiation problems*, Eng. Anal. Bound. Elem., 32 (2008), pp. 196–209.